Computational Intelligence: Neural Networks and Kernel Methods

Bernardete Martins Ribeiro

Coimbra

Março 2010

Sumário do seminário intitulado: "Computational Intelligence: Neural Networks and Kernel Methods"

Apresentado em cumprimento do disposto no art.n^{o} 8, alínea c), do Decreto-Lei n^{o} 239/2007 de 19 de Junho, tendo em vista a prestação de Provas de Agregação em Engenharia Informática, na Faculdade de Ciências e Tecnologia da Universidade de Coimbra, pela candidata Bernardete Martins Ribeiro.

Contents

Chapter 1

Introduction

1.1 Topic of the seminar

Computational Intelligence (CI) is an interdisciplinary field comprising Neural Networks, Fuzzy Systems, Evolutionary Algorithms and their hybrid paradigms. It is a methodology involving computing that exhibits an ability to learn and/or to deal with new situations, such that the system is perceived to possess intelligent behavior such as generalization, discovery, association and abstraction. CI is a wide concept that can be applied to a large number of fields, including but not limited to complex systems modeling, diagnosis, prediction, control and information processing. The main areas of application include computer science and informatics, engineering, finance, bioinformatics and medicine.

"Computational Intelligence: Neural Networks and Kernel Methods" was chosen to be the topic of this seminar since the main part of research work developed by the author has been in this area, as it is clearly shown by checking her publications. Beyond the presentation of the past and present research in computational learning, this seminar presents also the view of the author regarding the main future lines of the area.

This seminar is intended for researchers, and doctoral and master students that are interested in the topic. Previous knowledge of learning theory, modeling, decision making and optimization is not mandatory, but it would help to understand the contents presented.

Next section presents the scope of computational intelligence, namely, neural networks and kernel methods, as defined by the most relevant international organizations and scientific journals in the area. The chapter ends with a summary of the topics covered in this seminar, including the perspectives of future lines of research in the area.

1.2 Scope of Computational Intelligence: Neural Networks and Kernel Methods

The scope of computational intelligence is being widened continuously, well beyond the original concerns of learning algorithms that try to emulate human intelligence in computers. With two decades of history neural networks are powerful methods with a well-established theory. Kernel machines developed as the state-of-the-art of learning models are currently the cutting-edge methods for learning. Several international organizations, journals and conferences dealing closely with this concept have different definitions. Several journals deal with computational learning models including both approaches. The three most dedicated to the area are: IEEE Transactions on Neural Networks, journal of Machine Learning and journal of Neural Networks. Four conferences with rankings based on the general reputation of the conference in the field are also indicated. The scope of these organizations, journals and conferences is presented next.

1.2.1 International Organizations related to Computational Intelligence

Scope of the IEEE Computational Intelligence Society (CIS). The scope of CIS is defined in the site [1]:
"Theory, design, application and development of biologically and linguistically motivated computational paradigms emphasizing neural networks, connectionist systems, genetic algorithms, evolutionary programming, fuzzy systems, and hybrid intelligent systems in which these paradigms are contained."

Goals of International Neural Network Society/European Neural Network Society/ Japanese Neural Network Society (INNS/ENNS/JNNS). The goals of INNS/ENNS/JNNS are defined in the site [3]: *"Our goals to understand the information processes in the brain and to create more powerful brain-like machines for solving complex problems of the 21st century are very challenging indeed but with the joined efforts of all INNS members we will make a significant progress in the years to come."*

1.2.2 Scope of Scientific Journals strongly related to Computational Intelligence

IEEE Transactions on Neural Networks (TNN) scope. The scope of NN is defined in the site [2]. *"Devoted to the science and technology of neural networks, which disclose significant technical knowledge, exploratory developments, and applications of neural networks from biology to software to hardware. Emphasis is on artificial neural networks."*

Machine Learning (ML) Journal scope. The scope of ML is defined in the site [4]. *"Machine Learning is an international forum for research on computational approaches to learning. The journal publishes articles reporting substantive results on a wide range of learning methods applied to a variety of learning problems."*

Neural Networks (NN) Journal scope. The scope of NN is in the site [5]. *"Neural Networks is unique in its range and provides a forum for developing and nurturing an international community of scholars and practitioners who are interested in all aspects of neural networks and related approaches to computational intelligence. Neural Networks welcomes high quality articles that contribute to the full range of neural networks research, ranging from behavioral and brain modeling, through mathematical and computational analyses, to engineering and technological applications of systems that significantly use neural network concepts and algorithms. This broad range facilitates the cross-fertilization of ideas between biological and technological studies, and helps to foster the development of a new interdisciplinary community that is interested in biologically-inspired computational intelligence."*

1.2.3 Scope of Scientific Conferences related to Computational Intelligence

There is a large number of Conferences dedicated to the area of Computational Intelligence and it would not be possible to mention them all. A large group is dedicated to the core methods, algorithms and recent advances. Another group is focused on Computational Intelligence & Applications. The diversity and number of applications is very broad and - as referred above - it spans across all disciplines in engineering, computer science and informatics, finance, bioinformatics and medicine. I just mention those the autor is particular related with. They are ranked among 2nd and 3rd tier Conferences.

IEEE International Joint Conference on Neural Networks (IJCNN). The scope of this world event is completely defined in the annual Conference site. *IJCNN is the premier international conference on neural networks theory, analysis, and a wide range of applications. The conference theme in 2009 has been "The Century of Brain Computation Neural Network Alliances with Cognitive Computing and Intelligent Machine Embodiments." Studies into higher cognition and brain functions represent an ultimate frontier of scientific research. IJCNN2009 is a truly interdisciplinary event with a broad range of contributions on recent advances in neural networks, including neuroscience and cognitive science, computational intelligence and machine learning, hybrid techniques, nonlinear dynamics and chaos, various soft computing technologies, bioinformatics and biomedicine, and engineering applications.*

IEEE International Conference on Systems Man and Cybernetics (SMC). Aims and scope are indicated next. *SMC provides an international forum that brings together those actively involved in areas of interest to the IEEE Systems, Man, and Cybernetics Society, to report on up-to-the-minute innovations and developments, to summarize the state-of-the-art, and to exchange ideas and advances in all aspects of systems science and engineering, human machine systems, and cybernetics.*

International Conference on Neural Networks (ICANN). *It is an annual event, organized since 1991 by the European Neural Network Society (ENNS) in co-operation with the International Neural Network Society and the Japanese Neural Network Society. It provides a great discussion forum for the academic and industrial community on achievements not only related to artificial neural networks but to various other topics of artificial intelligence as well. More specifically, ICANN endeavors to address new challenges, share solutions and discuss future research directions in neural networks, learning systems, computational intelligence, and real-world applications. Moreover, it aims in promoting the significant benefits that can be derived by the use of neural networks and novel innovative applications are appreciated.*

International Conference on Adaptive and Natural Computing Algorithms (ICANNGA). Aims and scope are both defined in the Call for Papers and in the Conference site. *The aim of the conference is to bring together researchers from the same fields and to exchange and communicate new ideas, problems and solutions. The main areas of interest are: neural networks, evolutionary computation, soft computing, bioinformatics and computational biology, advanced computing, applications.*

1.3 Summary and outline

It can be observed from the previous section that Computational Intelligence (CI) methods cover a very broad area of research, and it is not easy to summarize what are these areas. Based on the purpose of CI, on the scopes of CI, ML, TNN, NN, top tier Conferences, and on the past and present contributions of the author to the field, two main areas of computational learning systems can be identified:

- Neural Networks
- Kernel Methods

This seminar presents a very brief state-of-the-art in these two areas. Several applications are presented for each area that result from the author's work, together with many other researchers. Note however that a much larger range of applications in the field of computational learning systems can be found in the literature. This seminar is organized into 5 chapters. The outline of the chapters and the main topics that each chapter considers are presented in the following.
In Chapter 2 neural networks are presented, tackling three architectures (Multi-layer Perceptron, Modular Networks and Hopfield Networks) with distinct goals meanwhile pursuing the final purpose of providing solutions for complex problems.
Kernel methods, namely, Support Vector Machines (SVMs) are described in Chapter 3 spanning three mathematical formulations (Hard margin, Soft margin and Nonlinear approach).
In Chapter 4 applications derived from the models presented previously are briefly described.
This seminar ends with Chapter 5 where the future developments of the computational learning systems are discussed. Future work towards the developement of whitening models are pointed as further lines of research.

Chapter 2

Neural Networks

Neural networks (NNs) are an abstraction of natural processes, more specifically of the functioning of biological neurons. Since early 1990s, they have been combined into a general and unified computational model of adaptive systems to utilize their learning power and adaptive capabilities. The two forms, learning and ability to adjust to new incoming patterns, create adaptation to a dynamic environment much more effective and efficient. Combined with evolutionary algorithms (EAs) they can be regarded as a general framework for adaptive systems i.e., systems that excel in changing their architectures and learning rules adaptively without human intervention.

2.1 Multilayer Neural Networks

MLP is one of the most extensively used neural networks. Given two sets of data, input/output pairs, MLP is able to develop a specific nonlinear mapping by adjusting the network weights by using a learning algorithm. It has been demonstrated that a two-layer MLP will adequately approximate any nonlinear mapping. The most used MLP training method is the backpropagation (BP) algorithm, where a steepest descent gradient approach and a chain-rule are adopted for back-propagated error correction from the output layer. Considerable efforts have been put into improving the speed of convergence, generalization performance, and the discriminative ability of MLP. To accelerate the BP algorithm, several heuristic rules have been proposed to adjust the learning rates or modify error functions [17]. Acceleration of training MLP can also be achieved by the use of other modifications to the standard BP algorithm, such as conjugate gradient BP, recursive least-square-based BP, and the Levenberg-Marquardt algorithm. To verify the generalization ability of MLP, the independent validation method can be used by dividing the available data set into a number of sub sets for training, validation and testing [16]. To improve the discriminative capability of MLP when applied to a classification task, a discriminative MLP learning rule was proposed which is more suitable for pattern classification tasks [13].

2.2 Modular Networks

There are strong biological and engineering evidences to support the fact that the information processing capability of NNs is determined by their architectures. Much work has been devoted to finding the optimal or near optimal NN architecture using various algorithms, including

EAs [55]. However, many real world problems are too large and too complex for any single ANN to solve in practice. There are ample examples from both natural and artificial systems that show that an integrated system consisting of several subsystems can reduce the total complexity of the entire system while solving a difficult problem satisfactorily. NN ensembles adopt the divide-and-conquer strategy. Instead of using a single large network to solve a complex problem, an NN ensemble combines a set of ANNs that learn to decompose the problem into sub-problems and then solve them efficiently. An ANN ensemble offers several advantages over a monolithic ANN [15]. First, it can perform more complex tasks than any of its components (i.e., individual ANNs in the ensemble). Second, it can make the overall system easier to understand and modify. Finally, it is more robust than a monolithic ANN, and can show graceful performance degradation in situations where only a subset of ANNs in the ensemble performs correctly. There have been many studies in statistics and ANNs that show that ensembles, if designed appropriately, generalize better than any single individuals in the ensemble. A theoretical account of why and when ensembles perform better than single individuals was presented in [52].

2.3 Hopfield Neural Networks

Associative memory networks include linear associative memory and Hopfield associative memory. Linear associative memory is an effective single-layer network for the retrieval and reduction of information. Given a key input pattern $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_K]$ and the corresponding output $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_K]$, associative memory learns the memory matrix $\mathbf{W}$ to map the key input $\mathbf{x}_i$ to the memorized output $\hat{\mathbf{y}}_i$. There are a number of ways to estimate the memory matrix. One estimate of the memory matrix $\mathbf{W}$ is the sum of the outer product matrices from pairs of key input and memorized patterns

$$\mathbf{W} = \sum_{k=1}^{K} \mathbf{y}_k \mathbf{x}_k^T. \tag{2.1}$$

To further reduce the memorized error, an error correction approach has been introduced to minimize the error function

$$E(\mathbf{W}) = \frac{1}{2} ||\mathbf{y}_k - \mathbf{W}\mathbf{x}_k||^2. \tag{2.2}$$

Hopfield associative memory is a nonlinear content-addressable memory for storing information in a dynamically stable environment [12]. The Hopfield network is a single-layer recurrent network which contains feedback paths from the output nodes back into their inputs. Given an input $\mathbf{x}(0)$, the Hopfield network iteratively updates the output vector by

$$\mathbf{x}(k+1) = f(\mathbf{W}\mathbf{x}(k) - \theta), \tag{2.3}$$

until the output vector become constant, where $f(\cdot)$ is the activation function. Associative memory is able to deduce and retrieve the memorized information from possibly incomplete or corrupted data.

Chapter 3

Kernel Methods

The concept of kernels was brought to the machine learning scenario by [9], introducing the technique of support vector machines (SVMs). Since then there has been considerable interest in this topic [42, 44]. Kernel methods are properly motivated theoretically and exhibit good generalization performance on many real-life data sets. The most popular kernel methods are support vector machines (SVMs) [54] and the recently introduced relevance vector machines (RVMs) [53, 8]. Kernel approaches are mostly binary while certain pattern classification applications are multi-label, multiclass problems. In this case, a problem with a set of $|\mathcal{C}|$ categories, $\mathcal{C} = \{c_1, c_2, \ldots, c_j, \ldots, c_{|\mathcal{C}|}\}$ is usually tackled as $|\mathcal{C}|$ independent binary classification problems under $\{c_j, \overline{c}_j\}$, for $j = 1, \ldots, |\mathcal{C}|$. In this case, a classifier for $\mathcal{C}$ is thus actually composed of $|\mathcal{C}|$ one-against-all binary classifiers.

In the following section we will detail one of the most accepted kernel methods, namely SVMs, treatead as a solution to a binary problem of classifying a set of data points $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_{|\mathcal{D}|}\}$ into two possible target classes: $\{c_j, \overline{c}_j\}$.

Kernel methods can be cast into a class of pattern recognition techniques in which the training data points, or at least a subset of them, are kept and used also during the prediction phase. Moreover, many linear parametric models can be transformed into an equivalent dual representation in which the predictions are based on a linear combination of a kernel function evaluated at the training data points [7]. For models which are based on fixed nonlinear feature space mapping, $\phi(\mathbf{x})$, the kernel function is given by the relation

$$K(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2), \tag{3.1}$$

making the kernel a symmetric function of its arguments so that $K(\mathbf{x}_1, \mathbf{x}_2) = K(\mathbf{x}_2, \mathbf{x}_1)$ [7]. The simplest example of a kernel function is the linear kernel, obtained by considering the identity mapping for the feature space in (3.1), so that $\phi(\mathbf{x}) = \mathbf{x}$, in which case $K(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2$.

The concept of a kernel formulated as an inner product in a feature space allows us to build interesting extensions of many well known algorithms by making use of the kernel trick, also known as kernel substitution. The general idea is that, if we have an algorithm formulated in such a way that the input $\mathbf{x}$ enters only in the form of scalar products, then we can replace the scalar product with some other choice of kernel. For instance, the technique of kernel substitution can be applied to principal component analysis (PCA) in order to develop a nonlinear version of PCA, KPCA (kernel-PCA) [43].

3.1 Support Vector Machines

Support Vector Machines (SVMs) were introduced by Vapnik [54] based on the structural risk minimization (SRM) principle, as an alternative to the traditional empirical risk minimization (ERM) principle. The main idea is to find a hypothesis h from a hypothesis space H for which one can guarantee the lowest probability of error $Err(h)$. Given a training set of n examples:

$$(\mathbf{x}_1, t_1), \dots, (\mathbf{x}_n, t_n), \mathbf{x}_i \in \mathbb{R}^N, t_i \in \{-1, +1\}, \tag{3.2}$$

SRM connects the true error of a hypothesis h with the error on the training set, Err_{train}, and with the complexity of the hypothesis (3.3):

$$Err(h) \leq Err_{train}(h) + O\left(\frac{v \ln(\frac{\eta}{v}) - \ln(\eta)}{n}\right). \tag{3.3}$$

This upper bound holds with a probability of at least $1 - \eta$. v denotes the VC-dimension[1] [54], which is a property of the hypothesis space H and indicates its expressiveness. This bound reflects the tradeoff between the complexity of the hypothesis space and the training error , i.e a simple hypothesis space will not approximate the desired functions, resulting in high training and true errors. On the other hand, a too-rich hypothesis space, i.e. with large VC-dimension, will result in overfitting. This problem arises because, although a small training error will occur, the second term in the right-hand side of (3.3) will be large. Hence, it is crucial to determine the hypothesis space with the sufficient complexity. In SRM this is achieved by defining a nested structure of hypothesis spaces H_i, so that their respective VC-dimension v_i increases:

$$H_1 \subset H_2 \subset H_3 \subset \dots \subset H_i \subset \dots \quad and \quad \forall i : v_i \leq v_{i+1} \tag{3.4}$$

This structure is *apriori* defined to find the index for which (3.3) is minimum. To build this structure the number of features is restricted. SVMs learn linear threshold functions of the type:

$$h(\mathbf{x}) = sign(\boldsymbol{\omega}.\mathbf{x} + b) = \begin{cases} +1 & \text{if } \boldsymbol{\omega}.\mathbf{x} + b > 0 \\ -1 & \text{otherwise,} \end{cases} \tag{3.5}$$

where $\boldsymbol{\omega} = (\omega_1, \omega_2, \dots, \omega_N)^T$ are linear parameters (weights) of the model and b is the bias. Linear threshold functions with N features have a VC-dimension of N+1 [54].

3.1.1 Linear hard-margin SVMs

In the simplest SVM formulation, a training set can be separated by at least one hyperplane, i.e. data are linearly separable and a linear model can be used:

$$y(\mathbf{x}) = \boldsymbol{\omega}^T \mathbf{x} + b. \tag{3.6}$$

In this case, SVMs are called linear hard-margin and there is a weight vector $\boldsymbol{\omega}'$ and a threshold b' that correctly define the model. Basically there is a function of the form (3.6) that satisfies

[1]Vapnik-Chervonenkis dimension

$y(\mathbf{x}_i) > 0$ for samples with $t_i = 1$ and $y(\mathbf{x}_i) < 0$ for points having $t_i = -1$, so that for each training data point $(\mathbf{x}_i, t_i)$

$$t_i y(\mathbf{x}_i) = t_i(\boldsymbol{\omega}'.\mathbf{x}_i + b') > 0. \tag{3.7}$$

As a rule there can be multiple hyperplanes that allow such separation without error (see Figure 3.1), and the SVM determines the one with largest margin ρ, i.e. furthest from the hyperplane to the closest training examples. The examples closer to the hyperplane are called support vectors (SVs) and have an exact distance of ρ to the hyperplane. Figure 3.2 depicts an optimal separating hyperplane and four SVs.

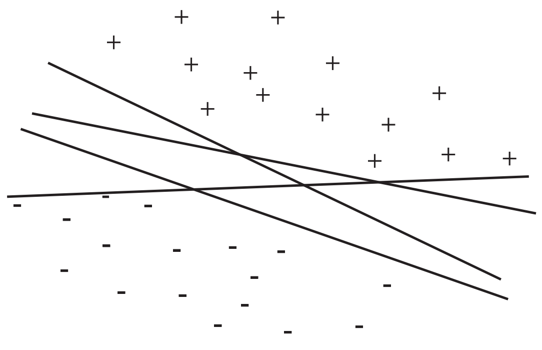

Figure 3.1: Possible hyperplanes separating positive and negative training examples.

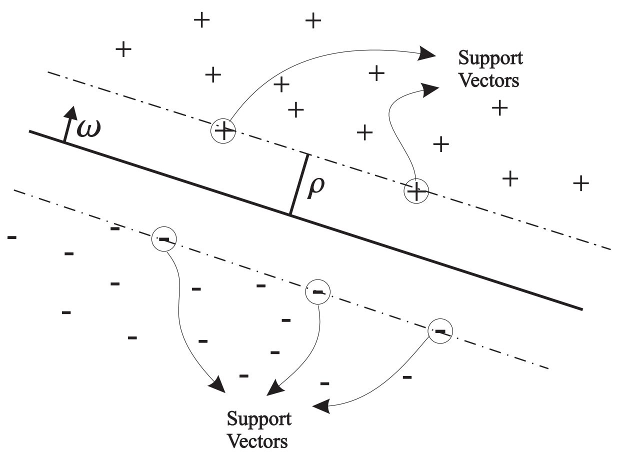

Figure 3.2: SVM optimal separating hyperplane and separating margin, ρ.

The perpendicular distance of a point from a hyperplane defined by $y(\mathbf{x}) = 0$, where $y(\mathbf{x})$ takes the form of (3.6), is given by $\frac{|y(\mathbf{x})|}{||\boldsymbol{\omega}||}$ [7]. As we are only interested in solutions for which all samples are correctly classified, so that $t_i y(\mathbf{x}_i) > 0$, the distance of a single data point $\mathbf{x}_i$ to the decision surface is given by

$$\frac{t_i y(\mathbf{x}_i)}{||\boldsymbol{\omega}||} = \frac{t_i(\boldsymbol{\omega}^T.\mathbf{x}_i + b)}{||\boldsymbol{\omega}||}. \tag{3.8}$$

The margin is then given by the perpendicular distance to the closest point $\mathbf{x}_i$ of the data set, and we wish to optimize the parameters $\boldsymbol{\omega}$ and b in order to maximize this distance. Thus, the maximum margin solution is found by solving

$$\arg\max_{\boldsymbol{\omega},b}\left\{\frac{1}{||\boldsymbol{\omega}||}\underset{i}{min}[t_i(\boldsymbol{\omega}^T\mathbf{x}_i + b)]\right\}. \tag{3.9}$$

Using a canonical representation of the decision hyperplane, all data points satisfy

$$t_i(\boldsymbol{\omega}^T \mathbf{x}_i + b) \geq 1. \quad i = 1, \dots, n \tag{3.10}$$

When the above equality holds, the constraints are said to be active for that data point, while for the rest the constraints are inactive. There will be always at least one active constraint, since there will always be a closest point, and once the margin has been maximized, there will be at least two active constraints. Then, the optimization problem consists of maximizing $||\boldsymbol{\omega}||^{-1}$, which is equivalent to minimizing $||\boldsymbol{\omega}||^2$, and so we have to solve the (primal) optimization problem:

$$\begin{aligned} \text{minimize: } & \frac{1}{2}\boldsymbol{\omega}.\boldsymbol{\omega}, \\ \text{subject to: } & \forall_{i=1}^{n} : t_i \left[\boldsymbol{\omega}.\mathbf{x}_i + b\right] \geq 1 \end{aligned} \tag{3.11}$$

These constraints establish that all training examples should lie on the correct side of the hyperplane. Using the value of 1 on the right-hand side of the inequalities enforces a certain distance from the hyperplane:

$$\rho = \frac{1}{||\boldsymbol{\omega}||}, \tag{3.12}$$

where $||\boldsymbol{\omega}||$ denotes the L_2-norm of $\boldsymbol{\omega}$. Therefore minimizing $\boldsymbol{\omega}.\boldsymbol{\omega}$ is equivalent to maximizing the margin. The weight vector $\boldsymbol{\omega}$ and the threshold b describe the optimal (maximum margin) hyperplane.

Vapnik [54] showed there is a relation between the margin and the VC-dimension. Considering the hyperplanes $h(\mathbf{x}) = sign(\boldsymbol{\omega}.\mathbf{x}+b)$ in an N-dimensional space as a hypothesis, if all examples $\mathbf{x}_i$ are contained in a hyper-circle of diameter R, it is required that, for examples $\mathbf{x}_i$:

$$|\boldsymbol{\omega}.\mathbf{x}_i + b| \geq 1 \tag{3.13}$$

and then this set of hyperplanes has a VC-dimension, v, bounded by:

$$v \leq min\left(\left[\frac{R^2}{\rho^2}\right], N\right) + 1, \tag{3.14}$$

where $[c]$ denotes the integer part of c. Thus, the VC-dimension is smaller as the margin becomes larger. Moreover, the VC-dimension of the maximum margin hyperplane does not necessarily depend on the number of features, but on the Euclidean length $||\boldsymbol{\omega}||$ of the weight vector optimized by the SVM. Intuitively this means that the true error of a separating maximum margin hyperplane is close to the training error even in high-dimensional spaces, as long as it has a small weight vector.

The primal constrained optimization problem (3.11) is numerically difficult to handle. Therefore, one introduces Lagrange multipliers, $\alpha_i \geq 0$, with one multiplier for each of the constraints in (3.11), obtaining the Lagrangian function

$$L(\boldsymbol{\omega}, b, \boldsymbol{\alpha}) = \frac{1}{2}||\boldsymbol{\omega}||^2 - \sum_{i=1}^{n} \alpha_i[t_i(\boldsymbol{\omega}^T \mathbf{x}_i + b) - 1], \tag{3.15}$$

where $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n)^T$. Note the minus sign in front of the Lagrangian multiplier term, because we are minimizing with respect to $\boldsymbol{\omega}$ and b, and maximizing with respect to $\boldsymbol{\alpha}$. Setting the derivatives of $L(\boldsymbol{\omega}, b, \boldsymbol{\alpha})$ with respect to $\boldsymbol{\omega}$ and b to zero, we obtain the following two conditions:

$$\boldsymbol{\omega} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i, \tag{3.16}$$

$$0 = \sum_{i=1}^{n} \alpha_i t_i. \tag{3.17}$$

Eliminating $\boldsymbol{\omega}$ and b from $L(\boldsymbol{\omega}, b, \boldsymbol{\alpha})$ gives the dual representation of the maximum margin problem

$$\begin{aligned} \text{maximize: } & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} t_i t_j \alpha_i \alpha_j (\mathbf{x}_i . \mathbf{x}_j) \\ \text{subject to: } & \sum_{i=1}^{n} t_i \alpha_i = 0 \\ & \forall i \in [1..n] : 0 \leq \alpha_i \end{aligned} \tag{3.18}$$

The matrix $\mathbf{Q}$ with $Q_{ij} = t_i t_j (\mathbf{x}_i . \mathbf{x}_j)$ is commonly referred to as the Hessian matrix. The result of the optimization process is a vector of Lagrangian coefficients $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^T$ for which the dual problem (3.18) is optimized. These coefficients can then be used to construct the hyperplane, solving the primal optimization problem (3.11) just by linearly combining training examples (3.19).

$$\boldsymbol{\omega}.\mathbf{x} = \left(\sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i \right) .\mathbf{x} = \sum_{i=1}^{n} \alpha_i t_i (\mathbf{x}_i \mathbf{x}) \qquad \text{and} \qquad b = t_{SV} - \boldsymbol{\omega}.\mathbf{x}_{SV} \tag{3.19}$$

Only support vectors (SVs) have a non-zero α_i coefficient. To determine b from (3.18) and (3.19), an arbitrary support $\mathbf{x}_{SV}$ vector with its class label t_{SV} can be used.

3.1.2 Soft-margin SVMs

Hard-margin SVMs fail when the training data are not linearly separable, since there is no solution to the optimization problems (3.11) and (3.18). Most pattern classification problems are not linearly separable, thus it is convenient to allow misclassified patterns, as indicated by the structural risk minimization [54]. The rationale behind soft-margin SVMs is to minimize the weight vector, like the hard-margin SVMs, but simultaneously minimize the number of training errors by the introduction of slack variables, ξ_i. The primal optimization problem of (3.11) is now reformulated as

$$\begin{aligned} \text{minimize: } & \frac{1}{2} \boldsymbol{\omega}.\boldsymbol{\omega} + C \sum_{i=1}^{n} \xi_i, \\ \text{subject to: } & \forall_{i=1}^{n} : t_i \left[\boldsymbol{\omega}.\mathbf{x}_i + b \right] \geq 1 - \xi_i \\ & \forall_{i=1}^{n} : \xi_i > 0. \end{aligned} \tag{3.20}$$

If a training example is wrongly classified by the SVM model, the corresponding ξ_i will be greater than 1. Therefore $\sum_{i=1}^{n} \xi_i$ constitutes an upper bound on the number of training errors. The factor C in (3.20) is a parameter that allows the tradeoff between training error and model complexity. A small value of C will increase the training errors, while a large C will lead to behavior similar to that of a hard-margin SVM.

Again, this primal problem is transformed in its dual counterpart, for computational reasons. The dual problem is similar to the hard-limit SVM (3.18), except for the C upper bound on the Lagrange multipliers α_i:

$$\begin{aligned} \text{maximize: } & \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} t_i t_j \alpha_i \alpha_j (\mathbf{x}_i . \mathbf{x}_j) \\ \text{subject to: } & \sum_{i=1}^{n} t_i \alpha_i = 0 \\ & \forall i \in [1..n] : 0 \leq \alpha_i \leq C \end{aligned} \tag{3.21}$$

As before, all training examples with $\alpha_i > 0$ are called support vectors. To differentiate between those with $0 \leq \alpha_i < C$ and those with $\alpha_i = C$, the former are called unbounded SVs and the latter are called bounded SVs.

From the solution of (3.21) the classification rule can be computed exactly, as in the hard-margin SVM:

$$\boldsymbol{\omega}.\mathbf{x} = \left(\sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i\right).\mathbf{x} = \sum_{i=1}^{n} \alpha_i t_i (\mathbf{x}_i \mathbf{x}) \qquad \text{and} \qquad b = t_{SV} - \boldsymbol{\omega}.\mathbf{x}_{SV} \tag{3.22}$$

The only additional restriction is that the SV $(\mathbf{x}_{SV}, t_{SV})$ for calculating b has to be an unbounded SV.

3.1.3 Nonlinear SVMs

Linear classifiers are inappropriate for many real-world problems, since the problems have inherently nonlinear structure. A remarkable property of SVMs is that they can easily be transformed into nonlinear learners [9]. The attribute data points $\mathbf{x}$ are mapped into a high-dimensional feature space X'. Despite the fact that the classification rule is linear in X', it is nonlinear when projected into the original input space. This is illustrated with the following example consisting of a vector $\mathbf{x} = (x_1, x_2)$, with two attributes x_1 and x_2. Let us choose the nonlinear mapping:

$$\Phi(\mathbf{x}) = \Phi((x_1, x_2)^T) = (x_1^2, x_2^2, \sqrt{2}x_1x_2, \sqrt{2}x_1, \sqrt{2}x_2, 1)^T \tag{3.23}$$

Although it is not possible to linearly separate the examples in Figure 3.3a), they become linearly separable in Figure 3.3b), after the mapping with $\Phi(\mathbf{x})$ into the higher dimensional feature space.

In general, such a mapping $\Phi(\mathbf{x})$ is inefficient to compute. There is a special property of SVMs that handles this issue. During both training and testing, it is sufficient to be able to compute dot-products in feature space, i.e. $\Phi(\mathbf{x}_i).\Phi(\mathbf{x}_j)$. For special mappings $\Phi(\mathbf{x})$ such dot-products can be computed very efficiently using kernel functions $K(\mathbf{x}_1, \mathbf{x}_2)$. If a function $K(\mathbf{x}_1, \mathbf{x}_2)$ satisfies Mercer's Theorem, i.e. it is a continuous symmetric kernel of a positive integer operator, it is guaranteed to compute the inner product of the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ after they have been mapped into a new space by some nonlinear mapping Φ [54]:

$$\Phi(\mathbf{x}_1).\Phi(\mathbf{x}_2) = K(\mathbf{x}_1, \mathbf{x}_2) \tag{3.24}$$

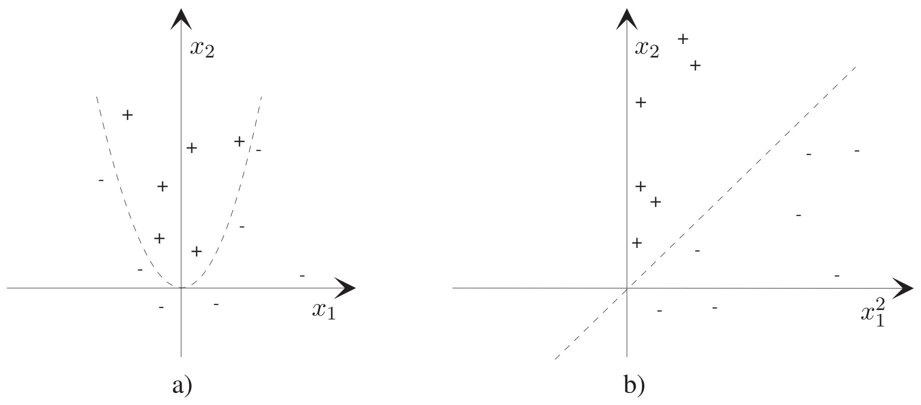

Figure 3.3: a) Nonlinearly separable training set in (w_1, w_2); b) Projection of the same set onto (x_1^2, x_2), where the set becomes linearly separable.

Depending on the choice of kernel function, SVMs can learn polynomial classifiers (3.25), radial basis function (RBF) classifiers (3.26) or two-layer sigmoid neural networks (3.27).

$$K_{poly}(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1.\mathbf{x}_2 + 1)^d \tag{3.25}$$

$$K_{RBF}(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\gamma(\mathbf{x}_1 - \mathbf{x}_2)^2\right) \tag{3.26}$$

$$K_{sigmoid}(\mathbf{x}_1, \mathbf{x}_2) = \tanh\left(s(\mathbf{x}_1.\mathbf{x}_2) + c\right) \tag{3.27}$$

The kernel $K_{poly}(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1.\mathbf{x}_2 + 1)^2$ corresponds to the mapping in Equation (3.23). To use a kernel function, one simply substitutes every occurrence of the inner product in equations (3.21) and (3.22) with the desired kernel function.

3.2 Kernel Parameters

For every algorithm making use of positive definite kernels there is the same question to answer: What is the best kernel for a given task? Ideally it should be set such that the resulting function minimizes the expected risk. Since this measure is not accessible, the most widely used approach is to approximate it using cross validation. When training Support Vector Machines (SVMs) two components need to be tuned: (1) the set of the kernel parameters whilst the other is (2) the regularization constant C controlling the tradeoff between the complexity of the function class and its ability to explain the data correctly. With cross validation, the parameters are picked from a range of values that perform the best on some held-out data.

Chapter 4

Computational Intelligence Models

4.1 Neural Networks

It was more than 50 years ago when John Von Neumann introduced his groundbreaking work on self-organization in cellular automata and on the intimate relationship between computers and the brain [24]. In the past decades, in particular since the rebirth and explosive development of the field in the mid 80s, neural network approaches have demonstrated their excellent potential to support fundamental theoretical and practical research towards new generations of artificial intelligence and intelligent computing. Practical models uphold a variety of applications in engineering and computer science, and also in many other fields such as finance, bioinformatics and medicine. In most of these areas, some examples are given for the methodologies (and applications) the author (in collaboration with other researchers) has worked with.

Multi-Layer Perceptron MLP

Intelligent signal processing is a major field where neural networks have clearly demonstrated their advantages over traditional statistical techniques. MLP is capable of adaptively approximating the function of any linear or nonlinear filter, and thus is very competitive when applied to data representation such as the extraction of efficient discriminative features, filtering, and enhancement of signals. Its capability of function approximation motivated our initial interest to use neural networks for problem solving in engineering. In [14] a theorem based on the seminal work of Kolmogorov for function approximation is proven. In addition, discussion of its impact to neural networks in the context of existence (and constructiveness) is provided. MLPs have been used in varieties of intelligent signal processing tasks, such as internal model control of a lime kiln [34, 33], fault detection and diagnosis in rotary lime kiln [35, 36], or in parts plastic industry [26, 22], For example, in data presentation, MLP has been successfully used in plastics image segmentation, feature location and parts defect detection in an injection molding machine [21]. An approach for finding the architecture of an MLP looking at the geometry of a given classification task was proposed in [36]. Pursuing a more algorithmic approach, a new architecture of an MLP-based neural network incorporating both hybrid learning and multi-topology has been developed in [23]. A new class designated Multiple Feed-Forward (MFF) networks, and the new gradient-based learning algorithm, Multiple Back-Propagation (MBP), are both proposed and analyzed. This new architecture (and algorithm) has a number of advantages over classical ones in that it can be trained to be noise-robust, and thus is more suitable for real-world environments in manufacturing industry [22]. Moreover, the methodology is very valuable in the pre-processing and feature extraction of engineering (and biomedical) data, and has been used in solving difficult signal processing and pattern recognition problems. Across the very successful applications we include information management [32], credit risk prediction [40] and medicine [37].

Modular Neural Networks

The feasibility of applying neural networks, specifically modular constructive neural networks, to mobile robot navigation was addressed in [48]. Due to their adaptive and generalization capabilities they are a reasonable alternative to more traditional approaches. The problem, in its basic form, consists of defining and executing a trajectory to a pre-defined goal while avoiding all obstacles, in an unknown environment. Some crucial issues arise when trying to solve this problem, such as an overflow of sensorial information and conflicting objectives. From this analysis, the necessity of introducing the concept of modularity has arisen, as a way to circumvent the existing conflicts. An original modular architecture instead of a monolithic approach is presented in [45] that effectively combines all the information available and navigates the robot through an unknown environment towards a goal position. In addition to modularity another concept is used - constructiveness - to find the best possible architecture [46]. The proposed architecture is successfully tested with the NOMAD 200^{TM} mobile robot.

Associative and Hopfield Neural Networks

Hopfield neural networks are proposed in [6] as an alternative to Dijkstra's shortest path algorithm for routing in Quality of Service (QoS) data networks. The proposed solution extends the traditional single-layer recurrent Hopfield architecture introducing a two-layer architecture that automatically guarantees an entire set of constraints held by any valid solution to the shortest path problem. This new method addresses some of the limitations of previous solutions, in particular the lack of reliability in what concerns successful and valid convergence. Experimental results show that an improvement in successful convergence can be achieved in certain classes of graphs. Additionally, computation performance is also improved at the expense of slightly worse results.

4.2 Kernel Methods

Kernel methods emerged among the most powerful tools available for a wide range of machine learning and more general function estimation problems. Kernels methods, broadly construed on Support Vector Machines (SVM) (and Relevance Vector Machines (RVM)), are widely considered as modern intelligence machines. They combine principles from statistics, optimization and learning in a sound mathematical framework able to solve large and complex problems in the areas of intelligent control and industry, information management, information security, finance and business, bioinformatics and medicine.

Intelligent Control and Industry

Intelligent signal processing aiming at quality monitoring, fault detection and control in an industrial process is performed by SVMs in [27, 31, 29] as part quality monitoring tools by analyzing complete data patterns. Assessment of the designed model is done by comparing it to RBF neural networks. Results show that SVM models with appropriately selected kernels present superior performance. This is even more marked when looking at the separability in the chosen high-dimensional feature space by SVM, which confirms that the kernel parameters indeed have an effect on the desired accuracy [28].

SVMs are able to handle feature spaces of high dimension and automatically choose the most discriminative features for estimation problems. In regression problems, more generalized forms based on some distance measure should be investigated. Under the assumption that Euclidean distance has a natural generalization in form of the Minkovsky distance function, the author proposed in [30] a new kernel function for model regression. It was tested in the setting of the Box-Jenkins furnace time series experiencing significant rise in the overall accuracy. In [10] the author together with other researchers successfully design a support vector regression based model in the sagittal plane for control of a biped robot.

Information Management

Computers do not “understand” the content of a book or a document, before retrieving, translating, recommending or summarizing it. Yet they exhibit (approximately) a behavior that we would consider intelligent. They do it by analyzing many examples of the appropriate behavior, and then learning to emulate it. Most successes depend on statistical pattern analysis and inductive learning inference based on a large number of pattern instances. In [51] a proposal for (and a review of) kernel techniques and approaches are presented in the scope of text classification, a crucial stage to information retrieval and modern search engines.
With the huge amount of information available in digital form, the computational intelligence models should be able to deal with (i) almost infinitely many unlabeled data (ii) to integrate learning models and (iii) to distribute tasks to solve the complex problems involved. The former issue is tackled in [49, 50] while for solving the latter two a distributed text classification with ensemble kernel-based learning approach is presented in [47].

Information Security

Steganography secretly hides information in digital products, increasing the potential for covert dissemination of malicious software, mobile code, or information. To deal with the threat posed by steganography, steganalysis aims at the exposure of the stealthy communications. A new scheme is proposed in [18] for steganalysis of JPEG images which, being the most common image format, is believed to be widely used for steganography purposes as there are many free or commercial tools for producing steganography using JPEG covers. In a similar motivated approach, described in [20], a set of features based on the image characteristics are extracted for model construction. Further, a new parameter - image complexity - is designed showing to enhance the model performance. Several nonparametric learning designs are built in [11] for resilient computational intelligence models in the same validation set of JPEG images.

Finance and Business

The large availability of financial data puts machine learning at the center stage giving rise to computational intelligence models. In response to the recent growth of the credit industry and to the world economic crisis early discovery of bankruptcy is of great importance to various stakeholders. Yet the rate of bankruptcy has risen and it is becoming harder to estimate as companies become more complex and the asymmetric information between banks and firms increases. The author in joint work has looked at several kernel models for financial distress prediction [40]. The significant results did not overlook the dimension reduction of the overall financial ratios. In fact, our recent focus on preprocessing the financial data led to very succeeded frameworks for evaluating firms financial status [41, 39, 38].

Bioinformatics and Medicine

In the 1990s genomic data started becoming available. Since mathematical models aimed at capturing the physics of transcription processes quickly proved to be unworkable, bioinformaticians turned to Computational Intelligence models for help in tasks such as gene finding and protein structure prediction. In [19] class prediction and feature selection, two learning tasks that are strictly paired in the search of molecular profiles from microarray data, were performed with SVMs. The models with RBF kernels have been shown to present a good choice, thus providing clues for cancer classification of individual samples. Recently, proteomic data considered potentially rich, but arguably unexploited, for genome annotation was used in [25]. The idea of using manifold (and supervised distance metric) learning for feature reduction combined with an SVM classifier of mass spectrometry was successful applied in biomedical diagnosis and protein identification. In [37], the author (in cojoint work) proposes a real-time predictor for Ventricular Arrhythmias (VA) detection using SVMs which shows favorable performances as compared to NN models.

Chapter 5

Concluding Remarks

Recent years have seen many new developments in computational intelligence (CI) techniques and, consequently, led to an exponential increase in the number of applications in a variety of areas, including: engineering, computer science, finance, and biomedical. Although this discipline is well-established, recent research shows that new algorithms, and the beauty of the subject, attracts every year newcomers to investigate in this area. The next steps are undoubtedly in discovering new paths to shaping these models with better explanation tools which might be able to communicate and transfer knowledge to the scientists in all areas.
This summary is not intended to be a in-dept analysis of all the problems behind computational learning models nor a broad coverage of the topic. It lays down the author's thought on many years of experience dealing with many problems of scientific interest.

Bibliography

[1] Computational Intelligence Society. http://ieee-cis.org/about_cis/.

[2] IEEE Transactions on Neural Networks. http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=72.

[3] International Neural Network Society. http://www.inns.org/.

[4] Machine Learning Journal. http://www.springer.com/computer/artificial/journal/10994.

[5] Neural Networks. http://www.inns.org/nnjournal.asp.

[6] F. Araújo, B. Ribeiro, and L. Rodrigues. A neural network for shortest path computation. *IEEE Transactions on Neural Networks*, 12(5):1067–1073, 2001.

[7] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[8] C. Bishop and M. Tipping. *Bayesian Regression and Classification, Advances in Learning Theory: Methods, Models and Applications*, volume 190 of *NATO Science Series III: Computer and Systems Sciences*, pages 267–285. IOS Press, 2003.

[9] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *ACM Annual Workshop on Computational Learning Theory - COLT 1992*, pages 144–152, 1992.

[10] J. Ferreira, M. Crisóstomo, P. Coimbra, and B. Ribeiro. Control of a biped robot with support vector regression in sagittal plane. *IEEE Transactions on Instrumentation and Measurement*, 58(9):3167–3176, 2009.

[11] R. Ferreira, B. Ribeiro, C. Silva, Q. Liu, and A. Sung. Building resilient classifiers for LSB matching steganography. In *IEEE World Congress on Computational Intelligence, Int Joint Conf on Neural Networks (IJCNN'08)*, pages 1562–1567, Hong Kong, China, 2008.

[12] J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *National Academy of Science USA*, 79(8):2554 –2558, 1982.

[13] B. Juang and S. Katagiri. Discriminative learning for minimum error classification. *IEEE Trans Signal Processing*, 40(12):3043–3054, 1992.

[14] A. Kovačec and B. Ribeiro. Kolmogorov's theorem: From algebraic equations and nomography to neural networks. In et al. R. F. Albrecht, editor, *Int Conf on Neural Networks*

and Genetic Algorithms (ICANNGA'93), pages 40–47, Innsbruck, Austria, 1993. Springer-Verlag.

[15] Ludmila I. Kuncheva. *Combining Pattern Classifiers - Methods and Algorithms.* Wiley, 2004.

[16] E. Levin, N. Tishby, and S.A. Solla. A statistical approach to learning and generalization in layered neural networks. *IEEE*, 78(10):1568–1574, 1990.

[17] Y. Li, K. Wang, and D. Zhang. Step acceleration based training algorithm for feedfoward neural networks. In *Int Conf Pattern Recognition (ICPR)*, pages 84–87, 2002.

[18] Q. Liu, A. Sung, M. Qiao, Z. Chen, and B. Ribeiro. An improved approach for steganalysis of jpeg images. *Information Sciences*, 180(9):1643–1655, 2010.

[19] Q. Liu, A. Sung, and B. Ribeiro. Comparison of gene selection and machine learning for tumor classification. In *3rd Int Conf on Informatics in Control, Automation and Robotics, Workshop on Signal Processing and Classification*, pages 13–22, Setubal, Portugal, 2006.

[20] Q. Liu, A. Sung, and B Ribeiro. Image complexity and feature mining for steganalysis of least significant bit matching steganography. *Information Sciences*, 178(1):21–36, 2008.

[21] N. Lopes and B. Ribeiro. A data pre-processing tool for neural networks (DPTNN) use in a moulding injection machine. In *Proc of the Second World Manufacturing Congress, WMC'99, Durham, England*, pages 357–361, 1999.

[22] N. Lopes and B. Ribeiro. Hybrid learning in a multi-neural network architecture. In *IEEE-INNS Int Joint Conf on Neural Networks (IJCNN'01)*, volume 4, pages 2788 – 2793, 2001.

[23] N. Lopes and B. Ribeiro. An efficient gradient-based learning algorithm applied to neural networks with selective actuation neurons. *Neural, Parallel & Scientific Computations*, 11(3):253–272, 2003.

[24] John von Neumann. *The computer and the brain.* Yale University Press, New Haven, CT, USA, 1958.

[25] B. Ribeiro Q Liu, A H. Sung and M Qiao. Classification of mass spectrometry data using manifold and supervised distance metric learning. In *Int Conf on Bio-Inspired Systems and Signal Processing*, pages 396–401, Porto, Portugal, 2009.

[26] B. Ribeiro. Fault detection in a thermoplastic molding injection process using neural networks. In *INNS-IEEE Int Joint Conf on Neural Networks (IJCNN'99)*, volume 5, pages 3352–3355, Washington, US, 1999.

[27] B. Ribeiro. Support vector machines and RBF neural networks for fault detection and diagnosis. In *IEEE 8th Int Conf on Neural Information Processing (ICONIP'01)*, volume 2, pages 873–878, Shangai, China, 2001.

[28] B. Ribeiro. Support vector machines in fault tolerance control. In D. Dubois, editor, *Computing Anticipatory Systems,* American Institute of Physics (AIP) Conf Proceedings, pages 458–467, 2002.

[29] B. Ribeiro. Computational intelligence in manufacturing quality control. *Prezeglad Elektrotechiniczny (Electrotecnical Review Journal)*, 80(4):286–290, 2004.

[30] B. Ribeiro. On the evaluation of Minkovsky kernel for SVMs. *Neural Parallel & Scientific Computations*, 13:77–90, 2005.

[31] B. Ribeiro. Support vector machines for quality monitoring in a plastic injection molding process. *IEEE Transactions on Systems, Man and Cybernetics SMC - Part C: Applications and Reviews*, 35(3):401–410, 2005.

[32] B. Ribeiro and A. J. Cardoso. Evaluation system for e-learning with pattern mining tools. In *IEEE Int Conf on Systems, Man and Cybernetics(SMC'08)*, pages 3051–3056, Singapore, 2008.

[33] B. Ribeiro and A. Dourado. *Lime Kiln Simulation and Control by Neural Networks*, pages 163–191. Elsevier Science Publishers, North-Holland, 1995.

[34] B. Ribeiro, A. Dourado, and E. Costa. A neural network based control of a simulated industrial lime kiln. In *INNS-IEEE Int Joint Conf on Neural Networks (IJCNN'93)*, volume II, pages 2037–2040, Nagoya, Japan, 1993.

[35] B. Ribeiro, A. Dourado, and E. Costa. Lime kiln fault detection and diagnosis by neural networks. In D. W. Pearson et al., editor, *Int Conf on Neural Networks and Genetic Algorithms (ICANNGA'95)*, pages 112–116, Alès, França, 1995. Springer-Verlag.

[36] B. Ribeiro, A. Kovačec, and A. Dourado. Example of finding the architecture of a BPNN for a pulp and paper engineering application looking at the geometry of a given classification task. In *Int Conf on Engineering Applications of Neural Networks (EANN'95)*, pages 343–351, Helsinki, Finland, 1995.

[37] B. Ribeiro, A. C. Marques, J. O. Henriques, and M. Antunes. Choosing real-time predictors for ventricular arrhythmias detection. *Int Journal of Pattern Recognition and Artificial Intelligence*, 21(8):1–15, 2007.

[38] B. Ribeiro, C Silva, A Vieira, and J Carvalho das Neves. Extracting discriminative features using non-negative matrix factorization in financial distress data. In M. Kolehmainen et al., editor, *Int Conf on Adaptive and Natural Computing Algorithms*, pages 537–547. Lecture Notes in Computer Science, LNCS 4432, Part II, Springer-Verlag, Berlin-Heidelberg, 2009.

[39] B. Ribeiro, A. Vieira, J. Duarte, C. Silva, J. C. das Neves, Q. Liu, , and A. H. Sung. Learning manifolds for bankruptcy analysis. In M. Köppen et al., editor, *Int Conf on Neural Information Processing*, pages 722–729. Lecture Notes in Computer Science, LNCS 5506, Part I, Springer-Verlag, Berlin-Heidelberg, 2009.

[40] B. Ribeiro, A. Vieira, and J. Neves. Sparse bayesian classifiers: Bankruptcy-predictors of choice? In *World Congress On Computational Intelligence, IEEE Int Joint Conf on Neural Networks (IJCNN'06)*, pages 3377–3381, Vancouver, Canada, 2006.

[41] B. Ribeiro, A. Vieira, and J. Neves. Supervised isomap with dissimilarity measures in embedding learning. In *Ibero-American Conf on Pattern Recognition, Progress in Pattern Recognition, Image Analysis and Applications*, pages 389–396. Lecture Notes in Computer Science, LNCS 5197, Springer-Verlag, Berlin-Heidelberg, 2008.

[42] B. Schölkopf, C. Burges, and A. Smola. *Advances in Kernel methods*, pages 1–15. MIT Press, 1999.

[43] B. Schölkopf, A. Smola, and K. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[44] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

[45] C. Silva, M. Crisóstomo, and B. Ribeiro. Monoda: A neural modular architecture for obstacle avoidance without knowledge of the environment. In *IEEE-INNS-ENNS Int Joint Conf on Neural Networks (IJCNN'00)*, volume 6, pages 334 –339, Italy, 2000.

[46] C. Silva, M. Crisóstomo, and B. Ribeiro. Angular memory and supervisory modules in a neural architecture for navigating NOMAD. In R. Neruda V. Kúrkova, N. Steele and M. Kárný, editors, *Int Conf on Neural Networks and Genetic Algorithms (ICANNGA01)*, pages 173–176, Prague, Czech Republic, 2001. Springer-Verlag.

[47] C. Silva, Uros Lotrič, B. Ribeiro, and Andrej Dobnikar. Distributed text classification with an ensemble kernel-based learning approach. *IEEE Transactions on Systems, Man and Cybernetics, SMC - Part C: Applications and Reviews*, 99:1–11, 2010.

[48] C. Silva and B. Ribeiro. Navigating mobile robots with a modular neural network. *Neural Computing & Applications Journal*, 1(3–4):200–211, 2003.

[49] C. Silva and B. Ribeiro. On text-based mining with active learning and background knowledge using SVM. *Journal of Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 11(6):519–530, 2007.

[50] C Silva and B. Ribeiro. Improving text classification performance with incremental background knowledge. In Marios Polycarpou et al., editor, *Int Conf on Artificial Neural Networks (ICANN)*, pages 923–931. Lecture Notes in Computer Science, 5768, Springer-Verlag, Berlin-Heidelberg, 2009.

[51] C. Silva and B. Ribeiro. *Inductive Inference for Large Scale Text Categorization: Kernel Approaches and Techniques*, volume 255. Springer, Heidelberg-Berlin, Germany, 2010.

[52] E. K. Tang, P. N. Suganthan, and X. Yao. An analysis of diversity measures. *Machine Learning*, 65:247–271, 2006.

[53] M. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–214, 2001.

[54] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer Verlag, 1995.

[55] X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87:1423 – 1447, 1999.